

DATAFED: WEB SERVICES-BASED MEDIATION OF DISTRIBUTED DATA FLOW

Rudolf B. Husar, Kari Höijärvi and Stefan R. Falke

Center of Air Pollution Impact and Trend Analysis, Washington University
One Brookings Dr. Campus Box 1124
St. Louis, MO 63130-4899

Abstract-DataFed is a distributed web-services-based computing environment for accessing, processing and rendering environmental data in support of air quality management and science. The flexible, adoptive environment facilitates the access and flow of atmospheric data from provider to users by enabling the creation of user-driven data processing value chains. The approach of DataFed is mediation between users and data providers. DataFed assumes the autonomous and spontaneous emergence of distributed data sources. It non-intrusively *wraps* datasets for access by web services. The mediator software, composed of web services, provides homogeneous data views (e.g. geo-spatial, time views) using a global multi-dimensional data model. Application software written using web services consists of data browsers and analysis tools for distributed AQ data. Currently DataFed serves as data gateway for user programs; web pages, GIS, science tools as well as for loosely coupled Service Oriented Architecture Applications. Its federated data pool consists of over 50 datasets and the tools have been applied in several air pollution projects.

I. INTRODUCTION

The vision of DataFed is to facilitate better air quality management and science by more effective use of relevant data. The specific goals of DataFed are:

1. Facilitate the access and flow of atmospheric data from provider to users.
2. Support the development of user-driven data processing value chains.
3. Participate in specific application projects

The specific approach used in DataFed is to provide mediation, services between data users and data providers. DataFed serves as data gateway for user programs; web pages, GIS and science tools. Specifically, DataFed assumes spontaneous emergence of data from autonomous providers. It non-intrusively *wraps* datasets for access by web services.

II. FEDERATED DATA SETS

The project now has access to an array of datasets that allow the basic documentation of air quality. The federation currently facilitates access to over 50 datasets. The datasets include both near real-time and historical

observations. It is anticipated that additional chemical, meteorological, emissions, and GIS datasets will be added to the DataFed collection in the near future. The near-real-time data are available within hours to a day following the observations. The time resolution of these near-real-time data ranges from an hour to once a day.

All the datasets are cataloged, accessed, and distributed through the Datafed.net infrastructure. In most cases these providers themselves are simply passing data through as part of the value-adding data processing chain. Similarly, Datafed.net is designed to pass the data to other applications such as the FASTNET project, and other value-adding activities. For some datasets, such as ASOS_STI, SURF_MET, VIEWS_CHEM, ATAD the data are accessed through special arrangements between Datafed and the provider. In other cases e.g. NEXTRAD, NAAPS, TOMS_AI, and HMS_Fire the data are accessed from the provider's website without special arrangements with the provider.

In distributed systems, such as DataFed, the data are maintained by their respective providers in their native environment. Users access the distributed data through the 'wrappers' and mediators provided as federation services. An additional federation service is caching, i.e. local storage or pre-calculation of frequently requested queries.

Numeric Data Caching. The above listed datasets are point monitoring data that are updated hourly or daily by their providers. The updates are generally individual files of spatial pattern containing the most recent measurements. This local storage is convenient for the incremental maintenance of the database but it is inefficient for data access, particularly for temporal data views. In DataFed the caching consists of densely packed numeric 'data cubes' suitable for fast and efficient queries for spatial and temporal views. The data cubes are updated hourly, daily or intermittently as dictated by the data availability and user-need. The daily cache updates is monitored through the cache status console. The time series show the number of valid stations for each hour/day in the dataset. A drop in number of stations indicates problems in data acquisition, transmission or caching.

Image Caching. The creation of some of the data views is time consuming. For example spatial interpolation and

contouring of monitoring data may take 20-30 seconds, which makes interactive data browsing impractical. For frequently used views, the data are pre-rendered and stored as images for fast retrieval and interactive browsing. The FASTNET browsing tools including the Analysts Consoles and Animator are set up to use the cached images. For a subset of views, the image caching is performed automatically following the update of the numeric data cubes. The image datasets are from distributed providers, the data are fetched from the provider and passed to the user on the fly.

III. MEDIATOR SERVICES

Mediator services allow the creation of homogeneous data views of the wrapped data in the federation (e.g. geo-spatial, time...). A data view is a user-specified representation of data accessible through DataFed. Data views consist of a stack of data layers, similar to the stack of spatial GIS data except that DataFed views can represent temporal and other dimensional pattern. Each data layer is created by chaining a set of web services, typically consisting of a **DataAccessService** which is followed by the **DataRenderService**. When all the data layers are rendered, and **Overlay** service merges all the layers into a single rendered image. Subsequent to the data access-render-overlay, the **MarginService** adds image margins and physical scales to the view. Finally, layer-specific **AnnotationService** can add titles and other user provided labels to the view. Below is a brief description and illustration of data views. Further detail can be found elsewhere.

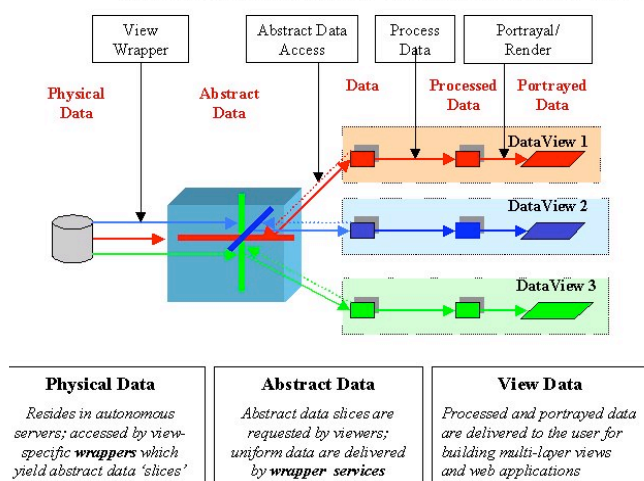


Figure 1. Generic data flow in DataFed services.

Views are defined by a 'view_state' XML file. Each view_state.xml contains the data and instructions to create a data view. Since the view creation is executed exclusively through web services, and web services are

'stateless' the data in the view_state.xml file represents the 'state' of the system, i.e. the input parameters for each of the web services used in the view. The view_state.xml file also contains instructions regarding the web service chaining. Thus, given a valid view_state.xml file an interpreter can read that file and execute the set of web services that creates a view. All data views, i.e. geo-spatial (GIS), temporal (time series), trajectories, etc are created in this manner.

The concept and the practical illustration data views is given below through specific examples. Suppose we have single layer view with a map data layer, margin and a title annotation, encoded in the file [TOMS_AI](#). The section on *Settings* contains parameters applicable for the entire view, such as *image_width* for the output image or *lon_min* for setting the zoom geo-rectangle. The section of *Services* specifies the services used to generate the data layers. In the example below, the combined **MapImageAccessRender** service accesses the map image and renders the boundaries. Normally there are more layers and services used in a view. The third section of each view state file is the *ServiceFlow*, which specifies the sequence and manner in which the individual services need to be executed.

A data view program can be executed by either the SOAP or the HTTP Get protocol. These illustrations use the simpler and transparent HTTP Get protocol in the form of a 'cgi' - URL. In order to execute the view generation program, the user (or her software agent) makes the following URL call (click on the URL below):

The entry after the ? view_state=TOMS_AI is the only user-specified parameter. The resulting view was generated dynamically by the web service interpreter at the DataFed website. Since the view file [TOMS_AI](#) had default values for all the services, it was able to generate an image of the view. However, each of the default parameters can be modified by the user by explicitly changing the desired parameters. These parameter changes are concatenated to the URL using the standard '&' character for instruction separation. In the example below, the parameters for image width, tom margin, minimum longitude, title text and location add the margin background color was changed. The difference in the two map renderings is evident.

http://webapps.datafed.net/dvoy_services/cgi.wsfl?view_state=TOMS_AI
http://webapps.datafed.net/dvoy_services/cgi.wsfl?view_state=TOMS_AI&lon_min=-60&lon_max=40&lat_min=0&text=ModifiedMap&y=center

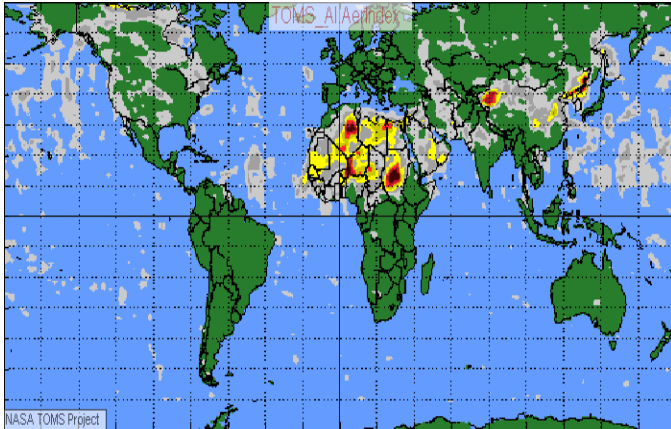


Fig. 2. Rendering of a view with default parameters.

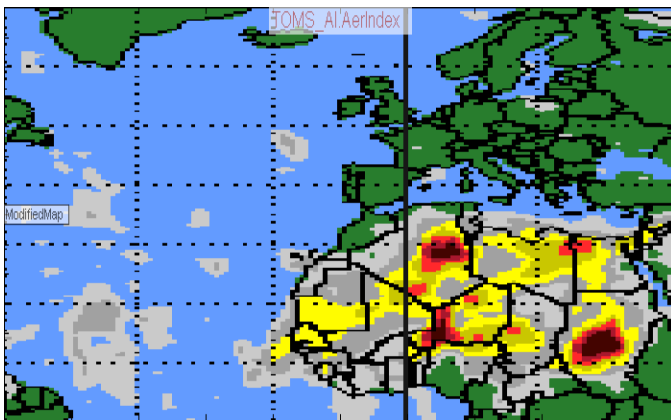


Fig. 3 Rendering of a view with user-modified parameters.

In summary, a user can take existing views and place those in her own web-page by calling a URL that defines the view. The user does not need to know the intricacies of the web service programming language, except the parameters that she wishes to modify in the view. This aspect of DataFed is similar to the the procedure for retrieving maps from standard OGC (Open GIS Consortium) servers. The preparation of data views is accomplished by the [DataFed view editor](#) tool.

Views can be simple or complex as shown in the current list of user-defined views page. Each illustrates a somewhat different aspect of the data or the DataFed services.

[List of VIEWS \(April 2004\)](#)

IV. WEB PROGRAMS AND PROJECTS

Specific user applications for data access, filtering and aggregation and rendering are built through web service composition. Since data views are images, the can be embedded into web pages as any other image. However, since the images are created dynamically, the content of the images can be controlled from the web page through a set of HTML controllers, connected to the image URL

through JavaScript or other scripting languages. For instance, the date of the displayed data in the view can be set by standard Text or Dropdown boxes, and the value of the user selection is used to set views URL. Such web pages, containing the dynamic views and the associated controllers, are called WebApps or MiniApps. By design, such WebApps web pages can be designed and hosted on the user's server. The only relationship to DataFed is that the view image are produced by the DataFed system. Unfortunately, full instructions for the implementation of WebApps is nut yet prepared. However, a number of WebApps have been prepared to illustrated the use of DataFed views in user's web pages. The JavaScript code of each WebApp is embedded in the source of each WebApp page.



Fig. 4. Typical web application program.

The DataFed 'middleware' aids the flow of aerosol data from the providers to the users and value-adders. Thus, DataFed neither produces, nor consumes data for its own purposes; it passes data from peer to peer. The data use, i.e. the transformation of the raw data to 'knowledge' occurs in separate, autonomous projects. However, the evolution of DataFed is fueled by projects and projects use its web-resources to be more effective. ([List of illustrative WebApps](#))

V. CONNECTION TO OTHER FEDERATIONS

The DataFed federation project strives to be a 'socially well-behaving' data integration endeavor. Beyond co-existence, it strives to cooperate, co-evolve, even merge with other data federations. DataFed is pursuing linkages to two major data federations, OGC and OPeNDAP. Both utilize standard web based HTTP protocols for data sharing and therefore suitable for easy integration with DataFed.

[OGC Specifications](#)

The Open Geospatial Consortium, Inc. ([OGC](#)) is a member-driven, not-for-profit international trade association that is leading the development of geoprocessing interoperability standards. OGC works with

government, private industry, and academia to create open and extensible software application programming interfaces for geographic information systems (GIS) and other geospatial technologies. [Adopted specifications](#) are available for the public's use at no cost.

The *Open-source Project for a Network Data Access Protocol*, [OPeNDAP](#), is a discipline-neutral protocol for requesting and providing data across the Web. The goal is to allow users, to access data in desired format for application software they already have. The protocol has its roots in the Distributed Oceanographic Data System (DODS), which successfully aided many researchers over the past decade. Ultimately, OPeNDAP is to support general machine-to-machine interoperability of distributed heterogeneous datasets.

VI. DATAFED SUMMARY

A. Software for the User

Data Catalog for finding and browsing the metadata of registered datasets.

Dataset Viewer/Editor for browsing specific datasets, linked to the Catalog.

Data Views - geo-spatial, time, trajectory etc. views prepared by the user.

Consoles, collections of views on a web page for monitoring multiple datasets.

Mini-Apps, small web-programs using chained web services (e.g. CATT, PLUME).

B. Software for the Developer

Registration software for adding distributed datasets to the data federation.

Web services for executing data access, processing and rendering tasks.

Web service chaining facility for composing custom-designed data views.

C. DataFed Technologies and Architecture

Form-based, semi-automatic, third-party wrapping of distributed data.

Web services (based web standards) for the execution of specific tasks.

Service Oriented Architecture for building loosely coupled application programs.

D. Issues

Reliability: Distributed computing issues: network reliability, bandwidth, etc

Chaining: Orchestrating distributed web services to act as a single application

Links: Linking users to providers and other federations (e.g. OGC, OPeNDAP)

Tech focus: Should it focus on data access, building web tools or both?

Domain focus: Is the focus on air quality appropriate? Other domains?

Sustainability: Can DataFed be self-sustaining supported by its community?

E. DataFed Project

The DataFed development was initiated by the CAPITA group: R. Husar (architecture), K. Höijärvi (software), S. Falke (data).

ACKNOWLEDGMENT

The initial funding by NSF (2001-4) grant: [Collaboration in Virtual Workgroups](#). NASA (2004-9) [Application of Satellite Data and Tools to PM Management](#) Funding from projects that use DataFed: NAM Emission, FASNET, CATT